Multimedia Tools and Applications
https://doi.org/10.1007/s11042-018-6573-5

A novel steganalysis method with deep learning for different texture complexity images

Shangping Zhong[1,2] • Cunmin Jia[1,2] • Kaizhi Chen[1,2] • Peng Dai[1,2]

Received: 13 June 2018 / Revised: 7 August 2018 / Accepted: 16 August 2018
Published online: 01 September 2018

Abstract
Most steganalysis methods based on deep learning don't distinguish the image texture complexity, but rather to mix all images for training. As a result, the differences of image content between images with different texture complexities are larger than the differences caused by steganographic signal, which is unfavourable for extracting the effective stegananalysis features. In order to reduce the influence of image content difference on steganalysis performance, this paper propose a steganalyais framework adopting the method of training model separately on dividing data sets. Firstly, according to the image texture feature, some correlated statistical features of gray level co-occurrence matrix are used as a measure index of texture complexity, and the data set is divided into several subsets of different complexity according to the index. Secondly, for images with different texture complexities, the corresponding steganalysis models are constructed and trained by using Most Effective Region (MER) and Inception ideas, respectively. For an image to be detected, its texture complexity is calculated first and the corresponding model is used for detection. Finally, an ensemble learning method is used to further improve the framework precision. The experimental results show that our proposed steganalysis method outperforms the handcrafted-features-based steganalysis methods and several CNN-based methods in detection accuracy.

Keywords Steganalysis · Deep learning · Texture complexity · Ensemble learning

✉ Shangping Zhong
spzhong@fzu.edu.cn

Cunmin Jia
N160320072@fzu.edu.cn

Kaizhi Chen
ckz@fzu.edu.cn

Peng Dai
fz_daipeng@163.com

Back Affiliation

1 Introduction

Information hiding is an important research direction in the field of information security. Its research and application focus on watermark copyright [5], data tampering and forensics [4] and privacy preserving [17]. In this paper we mainly study the steganalysis. Many new steganographic algorithms are content-adaptive and can be used to embed secret messages in the texture region or noise region of the image, reducing the efficiency of many conventional steganalysis methods. Therefore, the image texture is gradually considered in the steganalysis method. At present, there are two methods for steganalysis by using image textures. One is to extract statistical information of image texture as a feature for steganalysis. For example, Liu et al. [15] analyzed the advantages of the existing steganalysis algorithms and proposed an improved steganalysis algorithm based on local texture features in combination with the characteristics of the HUGO [21] steganographic algorithm. The algorithm used two local texture features, Local Ordinal Contrast Pattern (LOCP) and Local Phase Quantization (LPQ), and adopted the ensemble classifier to train and test model. However, due to relatively high feature dimension, imperfect performances limited the ability to further improve detection effects. Mao et al. [20] proposed a novel general steganalysis method based on cover image description. They made a complete quantitative description on cover image by extracting two novel features, texture feature and imaginary eigenvalue decomposition (IED) feature, avoided the curse of dimension and increased detection speed by a reducing dimension processor, and designed one classifier based on a flexible super-ellipse-sphere to enhance detection performance.

The other is to segment or classify the images by texture complexity, and extract features to train classifiers for different texture complexity images. Such as in paper [28], the author proposed a new spacial steganalysis method. Images were divided into image blocks with a size of S x S, and the given image was segmented according to the texture complexity feature of the sub-blocks. In the training phase, steganalysis features were extracted from each segmented subclass images to construct corresponding classifiers for training, and calculate the weight of each classifier according to the training accuracy. In the testing phase, the steganalysis features extracted from each sub image were sent to the corresponding classifier, and the judgment result of the whole image was obtained by weighting fusion. Tang et al. [27] proposed a content-adaptive steganalysis method to detect the WOW [10] steganographic algorithm. On account that WOW algorithm embedded secret messages into complex texture regions and noise regions with lower pixels modification loss and good visual quality, the author thought that the detection regions can be narrowed. Without analyzing the whole image, the author just analyzed features extracted from suspicious regions. By calculating the pixel distortion of stego image, one can locate the complex texture regions, and then extract the spatial rich model (SRM) [8] feature set to train a classifier for test. Through experiments, the author showed that for WOW algorithm, the proposed adaptive steganalysis method can obviously enhance analysis effect compared with the method to extract features from whole stego image. But for other steganographic algorithms, the author didn't further verify the efficiency of their detection methods. Juarez-Sandoval et al. [14] proposed an image adaptive steganalysis method against the Least Significant Bit Matching (LSBM) steganographic algorithm. This method analyzed the content characteristics of the input image to obtain the flat region and texture region. Through the embedding operation, the histogram of the difference matrix of the cover image in the flat region changes more than that in the texture region. In this case, only the flat region of the input image is considered to determine whether the image contains hidden information. Experimental results showed that this algorithm had good performance in detecting the traditional LSBM algorithm, and minimized the mismatch error of data sources.

Most of the aforesaid researchers use traditional methods of manually extracting features to train classifiers. Such methods rely on experience and domain knowledge when constructing features, and need special skills for feature selection, such as Ma et al. [19] used rough sets for attribute reduction to select features. More and more new adaptive steganographic algorithms [36], making many traditional methods' [18, 27] detection performance decline, and with the increase of feature dimension, the workload is also increasing.

In recent years, steganalysis methods based on deep learning have developed fast. Some research results show that a well-designed deep learning model can perform staganalysis tasks well. Qian et al. [23] proposed a CNN-based steganalysis architecture, which is the first time that deep learning has been applied in steganalysis. Unlike handcrafted-features-based methods, this CNN-based architecture merges feature extraction and classification into a single framework and automatically optimizes the parameters via back propagation. Compared with this architecture, the CNN-based model of Pibre et al. [22] has a smaller depth and larger width, uses a high-pass filter, comprises two convolutional layers, and removes pooling payer. Xu et al. [33] designed a new CNN-based steganalysis architecture that uses absolute value layer, batch normalization layer, convolutional layer (where the tanh activation function is applied in the first two convolutional layers) and global average pooling layer. With these modifications, this CNN-based architecture competes well with the rich model method in detecting S-UNIWARD [11]. Wu et al. [31] combined the deep residual network (DRN) [9] with steganalysis and found that shortcut and residual map can learn stego signals effectively. Therefore, these signals can be well preserved and do not decay or even diminish. In this case, DRN-based steganalysis methods can capture more effective features compared with CNN-based methods. Besides, some researchers mainly focused on the image preprocessing phase. Yuan et al. [35] used two high-pass filters with a size of 3×3 and a filter with a size of 5×5 to filter the input image to obtain three residuals, and then superposed the three residual images into a three-channel feature map for subsequent feature extraction. This method created higher detection accuracy compared with steganalysis network with only one high-pass filter. Ye et al. [34] used 30 high-pass filters in SRM to preprocess the input image, and Wu et al. [32] used 3 KV kernels to initialize the filters, but these filters should be updated with the network learning.

In this paper, we adopt the second method of using texture information, that is, dividing data set into several subsets according to the texture complexity and then constructing and training classifiers on these subsets. Because the deep learning methods in steganalysis have more advantages than traditional handcrafted-features-based steganalysis methods, we will use deep learning models for steganalysis. The main contributions of this paper are as follows:

(1) A texture complexity metrics based on the features of the gray level co-occurrence matrix is proposed. According to this metrics, the image data set is divided into two subsets with different texture complexities.
(2) For different texture complexity images, we use different ideas to construct several deep steganalysis networks to extract features and make classification. First, for simple texture images, we use the MER selection method [12] to obtain the region with largest embedding probability, reduce the network training cost, accelerate network training and finally improve detection accuracy. Second, for complex texture images, we utilize

30 high-pass filters to preprocess the image, and then draw on the idea of the famous Inception network [13], of using different sizes of convolutional kernels to obtain multiple sizes of pixel information, extract more abundant features so as to improve the classification performance of the network.

(3) We use the bagging ensemble method for each steganalysis network in (2) to further improve the detection accuracy of our proposed method.

The rest of this paper is organized as follows. In Section 2, we introduce the related techniques in this paper. In Section 3, we present the architectures of our proposed steganalysis networks for images with different texture complexities and correlation parameters in detail. In Section 4, we demonstrate the whole framework of our proposed steganalysis method. In Section 5, we show the impact of image texture complexity on steganalysis and compare our proposed method with the existing methods by conducting several experiments. In Section 6, we end this paper by delivering some concluding remarks.

2 Related techniques

2.1 The measurement of image texture complexity

For an image, although human can recognize the direction, thickness, and brightness of its texture, there is no uniform and effective metrics for quantifying texture complexity. Present methods first extract texture features and some eigenvalues are used to measure the complexity of the texture. The methods of extracting texture features mainly contain: (1) Statistical-based methods, such as differential value co-occurrence matrix and gray level co-occurrence matrix. Paper [3] uses uniform statistics based on the differential value co-occurrence matrix to measure the image texture complexity. Paper [30] uses the contrast value and entropy value of gray-level co-occurrence matrix as the metrics of image texture complexity. (2) Wavelet transform-based methods. These methods usually use the energy measure of the wavelet coefficients as texture features. However, the calculation of this index is relatively large, affecting the efficiency. (3) Model-based methods. Such as auto correlation model, random field model and fractal model.

Considering that the time efficiency of calculating the texture complexity can't be too low, and the texture complexity can be measured effectively, we draw on the idea in [30] and adopt the contrast and homogeneity of gray level co-occurrence matrix to determine the texture complexity of an image. Suppose the gray level of the image is N. $p(i,j)$ represents the number of times the element having the gray value i and the element having the gray value j in the gray level co-occurrence matrix appear in pairs in the case of the distance $(\Delta x, \Delta y)$.

(1) Contrast

$$G=\sum_{i=1}^{N}\sum_{j=1}^{N}(i-j)^2 p(i,j) \tag{1}$$

The contrast value reflects the degree of sharpness and texture depth of the image. The more pixels with larger contrast value are, the more complex the image is;

(2) Homogeneity

$$H = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{p(i,j)}{1 + (i - j)^2} \tag{2}$$

The homogeneity value reflects local change scope of image texture. A large value of homogeneity shows a lack of change in the texture between different regions of the image, so the image is very plain. On the contrary, there are more changes between different regions, so the texture is complex. In this paper, we use the following equation to represent the texture complexity:

$$C = \log(G + 1/H) \tag{3}$$

the usage of *log* function is to narrow the value scope.

2.2 Application of MER for steganalysis

Most of the state-of-the-art adaptive steganographic algorithms, for example HUGO, S-UNIWARD, WOW, and HILL [24], initially assign a distortion value to each pixel via a distortion function based on embedding cost before embedding information. Afterward, some advanced coding techniques, such as Syndrome-Trellis Codes (STCs) [6], are applied to minimize the expected distortion value for all pixels in texture areas and then pixels with maximum embedding probability (minimum distortion) are modified. Actually, the optimal embedding region obtained by an embedding algorithm may be applicable to other steganographic algorithms.

The MER selection method adopts the idea of adaptive steganography. Considering that different pixels have different contributions to steganalysis, instead of using all the pixels in the image as the input of feature extraction part, we find the complex texture region that hides information most likely as the input of subsequent network according to the characteristics of adaptive steganography, so the region is also called as most effective region (MER).

Although the size of input image is decreased, the region with the greatest contribution to image features extraction remains, and the rest of the image has a low probability or even no probability to be embedded, and its contribution to extracting steganalysis features is also very small. The features extracted from the most effective region are regarded as the principal component of features extracted from the whole image, which are more conducive to the steganalysis of images and improvement of detection efficiency. Figure 1 shows the idea of MER.

We use the embedding probability maps to find the MERs. First, we calculate the embedding probability of each pixel and then calculate and compare the sum of the embedding probability of all pixels in different regions. Second, we identify the region with the maximal probability sum as the MER. In an adaptive steganographic algorithm, the following distortion function of eq. (4) determines whether a pixel should be modified or not [11]:

$$D(X, Y) = \sum_{i=1, j-1}^{n,m} \rho_{i,j}\left(x_{i,j}, y_{i,j}\right) \tag{4}$$

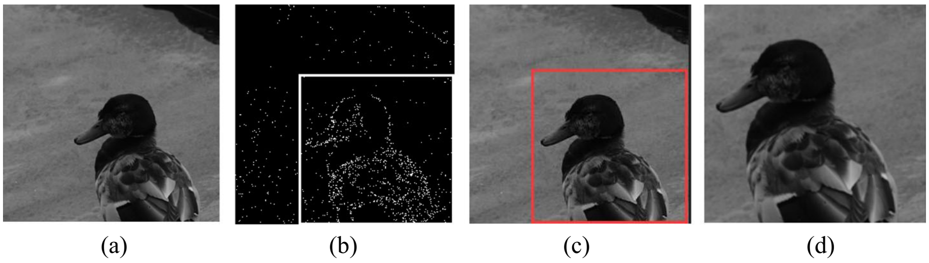


Fig. 1 The example of the MER. (a) is a stego image generated by embedding message to an cover image from BOSSbase data set. (b) shows the modification pixels after embedding. The region in red line has the most embedding probability. The region marked by red line in (c) is the MER of (a) obtained by (b). (d) is the final MER

This function is designed to measure the impact of embedding. Where$\rho_{i,j}$is the cost of pixel changes (from$x_{i,j}$to$y_{i,j}$). According to the distortion function, we can easily obtain the expected distortion and compute the probability map. The modification probability in [7] can be calculated as:

$$\pi_\lambda(Y) = \prod_{i=1,j=1}^{n,m} \frac{e^{-\lambda\rho_{i,j}\left(y_{i,j}\right)}}{\sum_{y_{i,j}\in\psi_{i,j}} e^{-\lambda\rho_{i,j}\left(y_{i,j}\right)}} \tag{5}$$

where λ must meet the following distortion constraints:

$$D_\varepsilon = \sum_{i=1,j=1}^{n,m} \sum_{y_{i,j}\in\psi_{i,j}} \pi_\lambda\left(y_{i,j}\right)\rho_{i,j}\left(y_{i,j}\right) \tag{6}$$

or

$$s = -\sum_{i=1,j=1}^{n,m} \sum_{y_{i,j}\in\psi_{i,j}} \pi_\lambda\left(y_{i,j}\right)\log\left(\pi_\lambda\left(y_{i,j}\right)\right) \tag{7}$$

2.3 Inception-like network

Thanks to the breakthrough of AlexNet in 2012, the trend of the mainstream network is that the network is deeper (the number of layers) and wider (the number of neurons). However, simply enlarging the network has some shortcomings: (1) too many parameters make it easy to overfit; (2) with a fixed training data set, the larger the network is, the greater the computational complexity will be. (3) the deeper the network is, the easier the gradient is disappeared (the gradient vanishing), and it is difficult to optimize the model. The original Inception v1 network [26], mixes convolutional kernels with sizes of 1 × 1, 3 × 3, 5 × 5 and pooling with a size of 3 × 3 (the 1 × 1 convolution operations are added for performance optimization), increasing network width and improving network adaptiveness. The main component of Inception v1 network is indicated as Fig. 2.

In this paper, we draw on the idea of Inception network structure, adopt convolutional kernels with sizes of 1 × 1, 3 × 3, 5 × 5 and 7 × 7 to extract different feature maps from the residual images after high-pass preprocessing, and then, concatenate the four feature maps for the input of next layer. Due to the usage of convolutional kernels with different sizes, the network can capture different sizes of pixels information, and obtain more abundant features, and hence improve the classification capacity. Figure 3 shows the Inception-like structure used in our model.

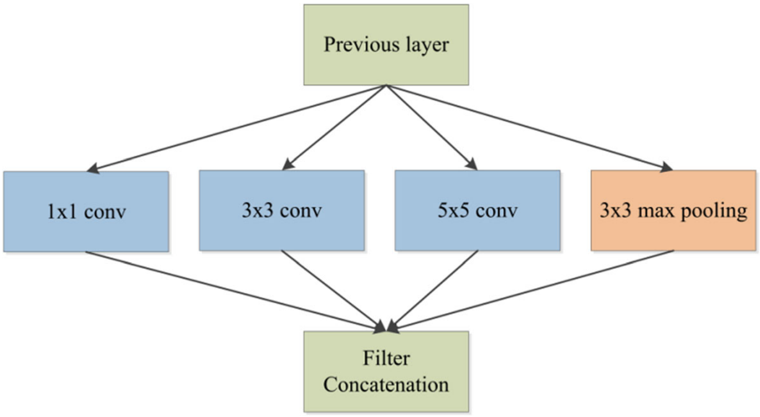


Fig. 2 The main component of Inception v1. The conv represents the convolutional operation

3 Steganalysis networks for different texture complexity images

3.1 Necessity of distinguishing texture complexities

In general, the effect of steganography on the statistical characteristic of image is less than that of image content [1, 2, 16]. The image is the Markov source with regional stability, and the images.

with same content complexity have similar pixel distribution characteristics. After embedding, secret information with strong randomness will produce different effects on the images with different statistical characteristics, and the difference of image content complexity can be reflected by the texture complexity [29]. Therefore, the image data set can be divided into several classes according to the image texture complexity, so that the statistical characteristics of each subclass of images are closer, steganalysis feature distributions are more aggregated, features of covers and stegos have better separability to reduce the effects of image content differences on the performance of steganalysis.

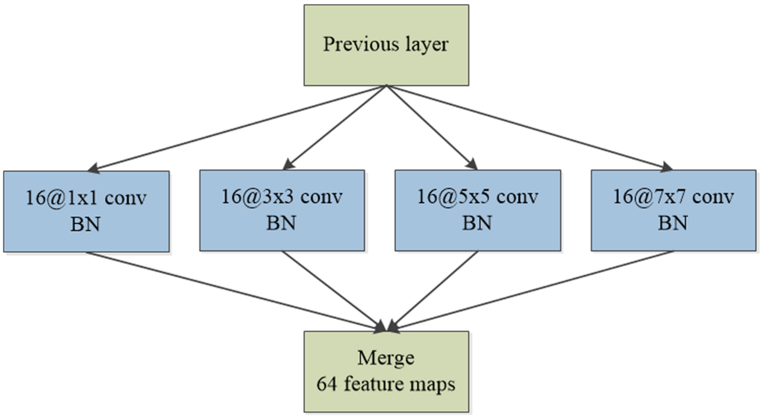


Fig. 3 The Inception-like structure used in our model. The conv and BN represent the convolutional and batch normalization operation

When we use deep network to extract features, if simple texture images and complex texture images are put into a same train batch, the learning efficiency will be decreased. Because the task of steganalysis is quite different from normal image content recognition task in computer vision (such as distinguishing cat and dog), it needs to learn the difference of tiny stego signals caused by embedding messages. If the images put into the same batch have significant differences of texture complexity, there may be difficulties to learn effective steganalysis features because image content difference is greater than that of the stego signal.

In addition, different texture complexity images have different sensitivity to secret information. Compared to the complex texture images, the embedding distortion of simple texture images is larger and is more likely to be detected by steganalysis methods, so it is necessary to analyze different texture complexity images separately.

Using deep learning techniques for steganalysis has a huge demand for hardware resources. Under the condition of guaranteeing detection accuracy, for simple texture images, we can construct relatively simple network, and for complex texture images we use relatively complex network.

3.2 Steganalysis network for simple texture images

For simple texture images, due to their relatively small complex texture regions, the effects of image content on the stego signal are relatively small compared with complex texture images, so the steganalysis is relatively simple. In this paper we use the deep steganalysis network shown in Fig. 4 to handle the simple texture images.

First, we use the MER selection method illustrated in section 2.2 to preprocess the input image, and then extract the region with most embedding probability, which is also the main texture region of the input image. The MER as the input of subsequent network can reduce the network computation complexity, accelerate the network training, and remain the most important part for steganalysis. Next, we use the KV high-pass kernel to filter the MER obtained in the first step to enhance the SNR between the steganographic signal and the image content and produce the noise residual. This step is noted as HPF (high-pass filtering). Then an activation function called TLU (Truncated Linear Unit) [34] will be used.

$$f(x)=\begin{cases}-T, & x<-T\\ x, & -T\le x\le T\\ T, & x>T\end{cases} \tag{8}$$

The TLU function is designed based on the characteristics of the steganographic signal, so it is effective to train the steganalysis network. After the TLU activation layer, there will follow a convolution layer (kernel size is 7×7), a batch normalization (BN) layer, a ReLU activation layer and a max pooling layer. Then, we adopt 4 Residual learning (ReSL) blocks in deep residual network in [31] to deepen the network and improve the feature extraction ability of network and remain the steganographic signal. Feature maps generated from previous layers are sent into a fully connected layer with 512 neurons, and mapped into 2 output neurons by a Softmax function. Finally, we get the probabilities of two classes.

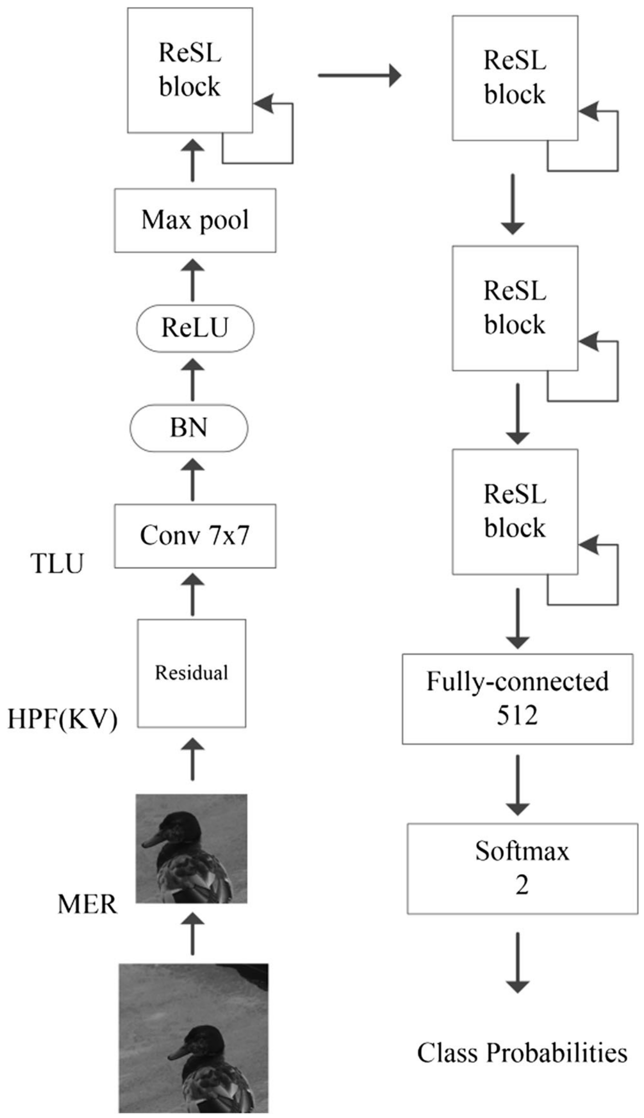


Fig. 4 Steganalysis network for simple texture images. The MER denotes the most effective region of input image, and ReSL denotes the residual learning block in deep residual network

3.3 Steganalysis network for complex texture images

For complex texture images, their local textures are very disorderly and change frequently. Tiny staganographic signals hiding in texture regions are difficult to be detected. Among traditional steganalysis methods, the SRM feature set has an optimal detection effect. It is the combination of different filter residual models that makes the rich model (RM) so successful. So we use the 30 basic high-pass filters used in calculating residual maps in SRM to generate different noise residuals of original image, including 7 residual classes in SRM: 8 filters in class "1", 4 in class "2", 8 in class "3", 1 in class "SQUARE

3 x 3", 1 in class "SQUARE 5 x 5", 4 in class "EDGE 3 x 3" and 4 in class "EDGE 5 x 5" with a maximum kernel size of 5 x 5. This step is also noted as HPF (high-pass filtering) like section 3.2.

The residual maps generated by high-pass filtering are stacked together, and activated by the TLU function used in section 3.2. Then we utilize the process structure of the Inception idea in section 2.3 to extract multiple sizes of features. The subsequent structure is similar to that of simple texture image steganalysis network. Fig. 5 shows the whole network structure.

3.4 Rationale for applying DRN for steganalysis

Fig. 6 shows a residual learning block. Where x is the input of the block, $F(x)$ is the residual mapping, c is the cover image, m is the weak stego signal generated by the embedded messages, and 0 is the zero signal. To detect the presence of secret information, the input image x must be accurately classified as:

$$\boldsymbol{x} = \begin{cases} \boldsymbol{c} + \boldsymbol{0}, \boldsymbol{cover} \\ \boldsymbol{c} + \boldsymbol{m}, \boldsymbol{stego} \end{cases} \tag{9}$$

With inputting x into a residual block, the identity mapping of the network puts forward c to the output of the block. The residual mapping fits 0 or m . Given that 0 and m are weak signals, they can be effectively learned by the residual mapping, and m can be captured by the residual network. Therefore, the weak stego signals are well preserved and enhanced through the network. This property makes DRN suitable for image steganalysis.

4 Our proposed steganalysis framework

To reduce the impact of image content difference on steganalysis and improve the performance of detection, in this paper we first use the metrics based on the gray level co-occurrence matrix statistical characteristics proposed in Section 2.1 to divide the image data set into 2 classes, simple texture and complex texture, separately adopt the MER selection method and Inception idea to construct steganalysis networks. Then we train two networks on two classes images to obtain two models. In test phase, we calculate the texture complexity of an image to be detected and load the corresponding model to decide whether it contains secret information.

4.1 Division of image data set

The image data set we used in our paper is the classical BOSSbase v1.01, which contains 10,000 Gy scale nature images with a size of 512×512. However, in deep learning task, small data set can not train the network well, so we augment the data set according to the method in [23]. Each image of 512×512 is cropped into 4 non-overlapping images of 256×256. Finally, a new data set including 40,000 cropped images is generated. In the subsequent sections, most follow-up experiments will base on this new data set. We use

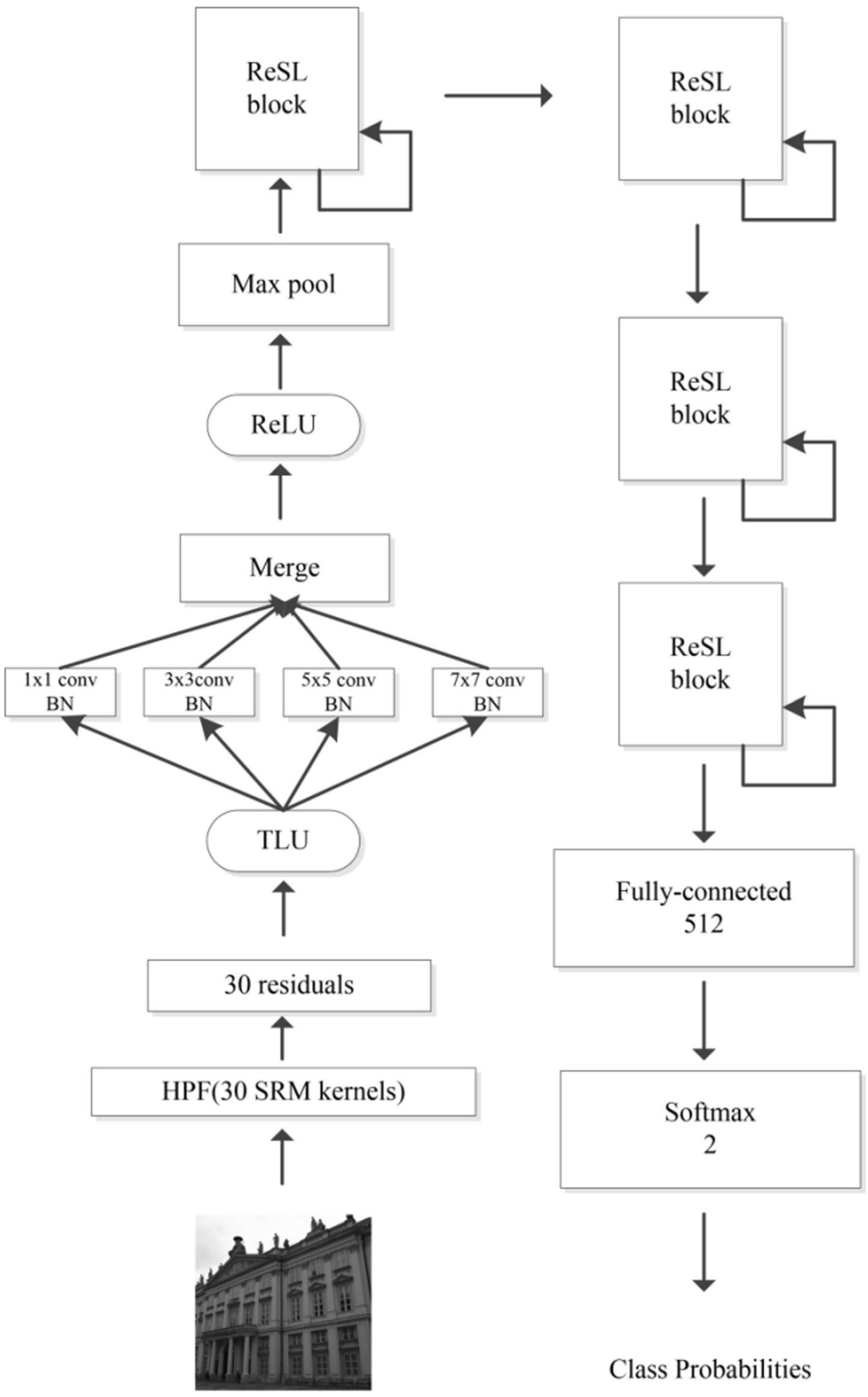


Fig. 5 Steganalysis network for complex texture images

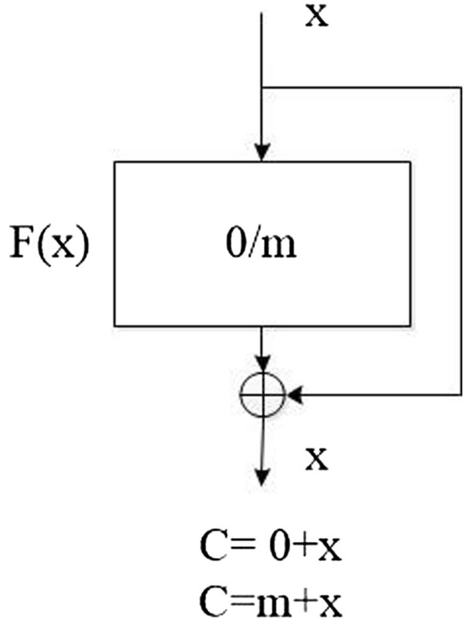


Fig. 6 Stego signal in a residual learning block

the metrics of texture complexity proposed in section 2.1 to calculate every image's texture complexity in new data set, and draw a frequency histogram to show the texture complexity distribution as Fig. 7.

We can see from Fig. 7 that the texture complexity of the whole data set mainly focuses on the range of 2 to 7. We use the equal area division method to split the whole data set into two parts: simple texture subset and complex texture subset. Each subset has 20,000 images for training and testing the network. Some examples in two subsets are shown in Fig. 8.

4.2 The framework of proposed steganalysis method

Figure 9 shows the whole framework of our proposed method, which consists of two parts: training process and testing process.

The training phase has two steps, First, split the training data set into simple texture subset and complex texture subset according to the texture complexity. Second, train two steganalysis networks for the two subsets in the first step. For simple texture images, we train the network proposed in section 3.2, which mainly contains the MER selection preprocessing layer, high-pass filtering layer (using the KV kernel), the TLU activation layer, 4 residual learning blocks and other components, like max pooling layer, fully-connected layers and Softmax layer. For complex texture images, we train the steganalysis network proposed in section 3.3, which mainly contains a high-pass filtering layer (using 30 linear kernels in SRM), the TLU activation layer, the Inception-like block, 4 residual learning blocks and so on. After training, we obtain two optimal models on two subsets for testing.

In the testing phase, for an image under detection, we should calculate its texture complexity first, then load corresponding trained models to predict the class labels. As a result, we can decide whether an image was embedded in Algorithm 1.

Algorithm 1 decide whether an image was embedded, if the image was embedded, return True else return False.

```
Input: an image I under detection .
Output: classification result.
1: #Steganalysis model for complex texture images
2: Function com_model(I)
3:     R=HPF(I)
4:     features =DRN(Inception(R))
5:     p1,p2=Softmax(features)
6:     If p1>p2 then
7:         Return True
8:     Else
9:         Return False
10:    End if
11:End function
12:
13: #Steganalysis model for simple texture images
14: Function sim_model(I)
15:     Mer=MER(I)
16:     R=HPF(Mer)
17:     features =DRN(R)
18:     p1,p2=Softmax(features)
19:     If p1>p2 then
20:         Return True
21:     Else
22:         Return False
23:     End if
24: End function
26: #Detecting whether an image was embedded with secret information
27: Function is_stego(I)
28:     Result ← False
29:     #Calculate the I's texture complexity C by using the following formula:
30:     $C = \log(G + 1/H)$
31:     If $C >$ threshold value then
32:         Load the trained steganalysis model for complex texture images
33:         result=com_model(I)
34:     Else
35:         Load the trained steganalysis model for simple texture images
36:         result=sim_model(I)
37:     End if
38:     Return result
39: End function
```

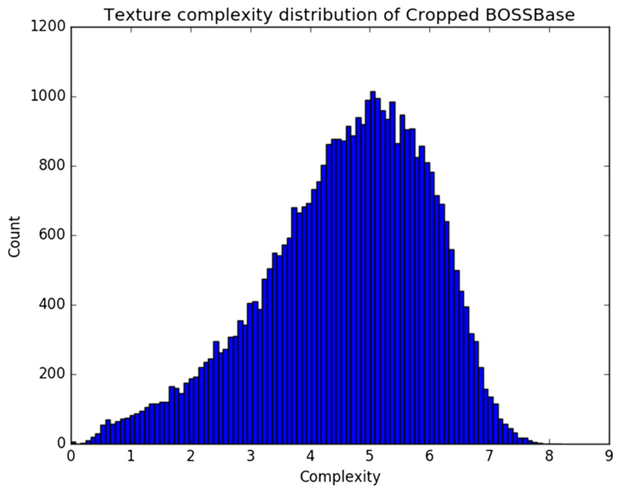


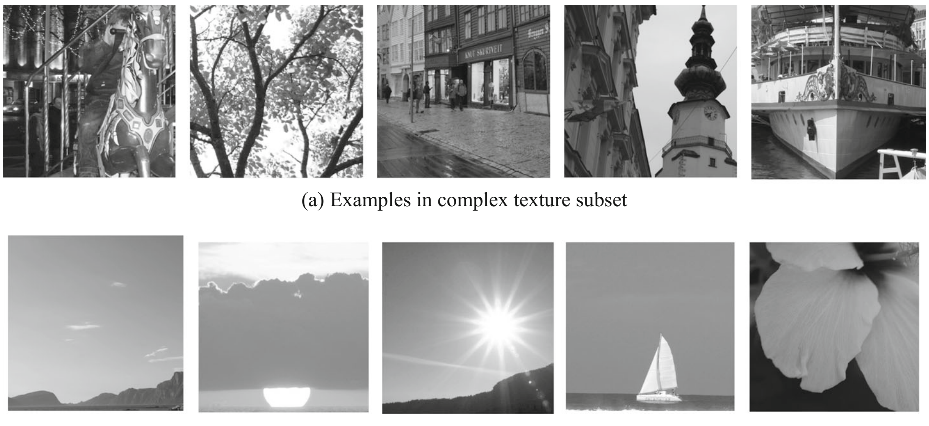
Fig. 7 The distribution of image texture complexity of the cropped data set

5 Experiments

In this section, we first conduct some experiments to study the impact of image texture complexity on steganalysis. Next, we validate the effectiveness of two steganalysis networks for two class images separately by several comparison experiments. Then, we validate the whole proposed steganalysis framework. Finally, we use the bagging ensemble method to improve the detection accuracy.

5.1 Experimental settings

The image data set used in this section consists of two parts. One is the cropped BOSSbase v1.01 data set with 40,000 images. This images set is used to train and test the two steganalysis networks. The other one is the BOSSbase v0.92 data set. This image set is used to validate the whole steganalysis framework.



(a) Examples in complex texture subset

(b) Examples in simple texture subset

Fig. 8 Sample images in two subsets with different image texture complexities

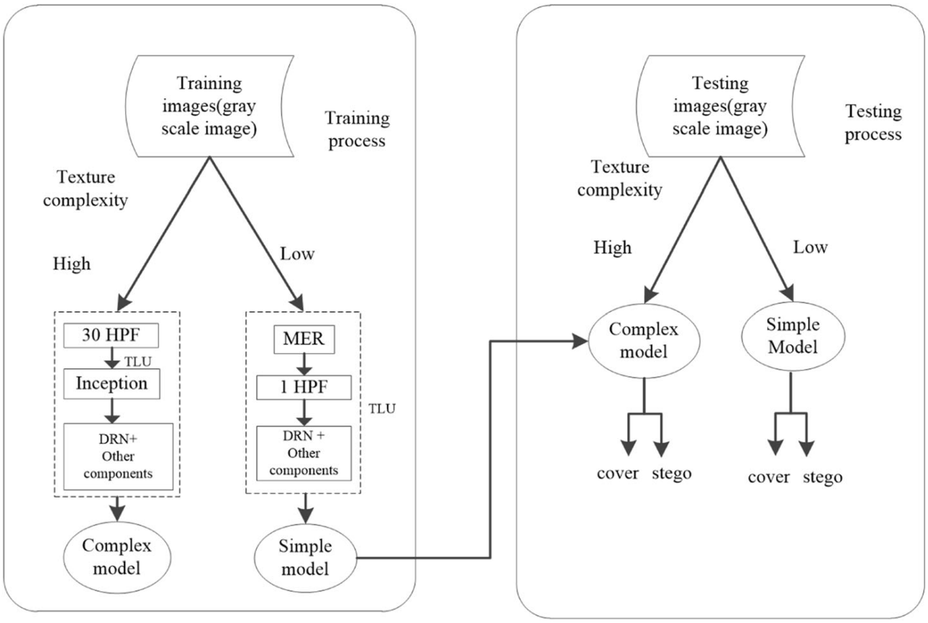


Fig. 9 The framework of proposed steganalysis method

For both steganalysis networks, in the residual learning and classification parts, we use zero mean Gaussian distribution with a standard derivation of 0.01 to initialize the weight matrices and use zero to initialize the biases vectors. The initial learning rateαstarts from 0.0001 and is divided by 10 per 50,000 training iterations. The training epochs have a maximal value of 100. The mini-batch stochastic gradient descent (SGD) optimizer is used to optimize the network parameters with a batch size of 20. The parameter of weight decay is set to 0.001. The T value in TLU activation function is set to 3. For simple texture image steganalysis network, the size of MER is 192 × 192. All of the experiments are conducted on the platform of tensorflow-gpu version 1.2.0. Two NVIDIA GTX 1080 Ti high-performance graphics cards are used to accelerate the training and testing.

5.2 The impact of image texture complexity on steganalysis

Split the 40,000 images of cropped BOSSbase v1.01 into two parts: simple texture images and complex texture images. Each part contains 20,000 images. We use the MATLAB version of WOW and S-UNIWARD steganographic algorithms to embed secret messages with an embedding rate of 0.4 bpp into two class texture complexity images and get corresponding 20,000 stego images. Fig. 10 shows how to divide training and testing data sets.

As the Fig. 10 shows, 10,000 covers and 10,000 corresponding stegos are randomly selected from the simple texture images as the training set, then 5000 covers and their corresponding stegos are randomly selected as the validation set. The remaining 5000 covers and their stegos are selected as the testing set. The division of complex texture images is the same as above. We use the deep residual steganalysis network (we call it DRSN in this paper) in [31] to conduct the validation experiments, which has the default setting in this paper. Table 1 shows the detection accuracy of the DSRN on different texture complexity images.

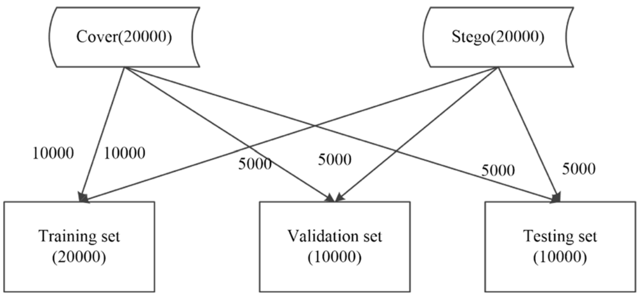


Fig. 10 The way to divide training and testing data sets

From Table 1, we can see that the detection accuracy is consistent with our intuition. The complex texture images have complex texture regions, so the image content has a great influence on steganalysis, and the stego signals are hard to be detected, resulting in the low detection accuracy. However, simple texture images has a high detection accuracy.

Fig. 11 shows the training speed of steganalysis network on both texture image sets and the training accuracy of each epoch.

We can see from the figure that the convergence speed of the model is different when training on images with two kinds of texture complexity. The speed of training model with simple texture images is faster, and it takes about 40 to 50 epochs of iteration to converge the network, the loss value will not change, and the accuracy is high. While training model with the complex texture images takes about 70 to 80 epochs of iteration, and when the model is stable, the training accuracy is lower than that of training model on simple texture images.

It can be seen that influence of image texture complexity on steganalysis can not be ignored, so we need to consider with different texture complexities separately, especially high texture complexity images, because in real life, images from social medias as well as the network are mostly have complex texture, images with pure color or flat texture are relatively few. Therefore, it is necessary to train the detection model for different texture complexity images to improve the performance of steganalysis.

5.3 The effectiveness of two steganalysis networks

In this section, we train and test several steganalysis methods on simple texture images and complex texture images separately and compare them with our proposed steganalysis method. These methods contain: (1) The SRM feature set which is the best in manually constructed feature sets combines the ensemble classifier (EC), abbreviated as SRM + EC. (2) The steganalysis network proposed by Qian in [23], abbreviated as Qian's network. (3) The steganalysis network proposed by Xu in [33], abbreviated as Xu's network. (4) The deep residual steganalysis network proposed in [31], abbreviated as

Table 1 The detection accuracies of trained models on different texture complexity images

Steganographic algorithms	Complex texture images	Simple texture images
WOW	82.1%	92.6%
S-UNIWARD	81.4%	90.0%

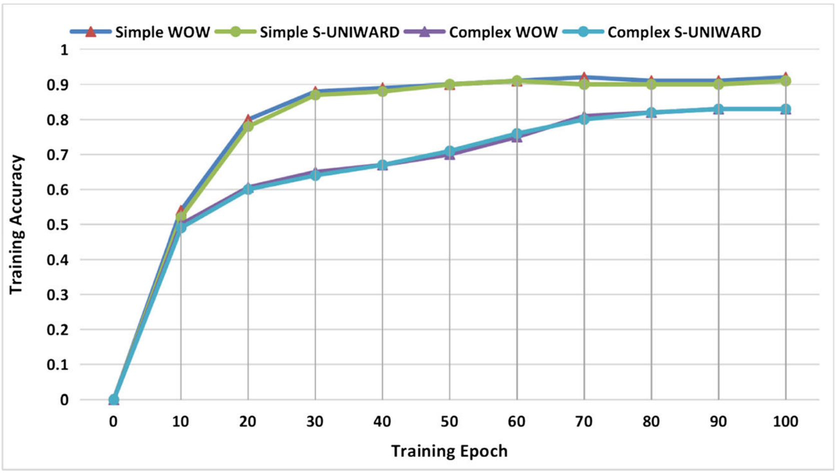


Fig. 11 Network training speed and accuracy on different texture complexity images

DRSN. We use 4 steganographic algorithms HUGO,WOW, S-UNIWARD and MiPOD [25] to embed secret messages into simple texture images and complex texture images with 0.4 bpp (bit per pixel) embedding rate to obtain stego images.

5.3.1 The effectiveness of steganalysis network for simple texture images

First, we train and test these steganalysis networks on simple texture images. For convenience, we mark our steganalysis network proposed for simple texture images as Sim_MER (because we mainly use the MER method in this case). The parameters in those 4 comparison methods follow the default settings in their papers. The parameters in our network follow the settings in section 5.1. Table 2 shows the comparison of testing accuracies of 5 methods. We can see from the Table 2, the testing accuracies on simple texture images are generally high. It is further illustrated that simple texture images are relatively easy to detect the existence of secret information. Comparing the first line with other lines, it can be seen that well-designed deep learning steganalysis methods, including the method of this paper, can be comparable to or even surpass the SRM + EC method. From the second, third, fourth and last lines, we find that the steganalysis method proposed in this paper has higher detection accuracy than other deep learning-based steganalysis methods for the usage of the deep residual learning and MER selection. So extracting the most effective region in a low texture complexity image can improve the network performance.

Table 2 The comparison of testing accuracies of 5 methods in simple texture case

Steganalysis Methods	HUGO	WOW	S-UNIWARD	MiPOD
SRM + EC	87.2%	85.7%	85.1%	84.8%
Qian's network	87.3%	86.9%	86.2%	85.1%
Xu's network	89.1%	88.6%	88.3%	87.3%
DRSN	92.6%	91.3%	90.9%	89.6%
Sim_MER(our method)	93.4%	92.7%	91.5%	90.7%

5.3.2 The effectiveness of steganalysis network for complex texture images

Similar to the previous notation in section 5.3.1, we note the steganalysis method for complex texture images as Com_Inception (because we mainly use the Inception-like structure in this network).

The parameters in those 4 comparison methods follow the default settings in their papers. The parameters in our network follow the settings in section 5.1. Table 3 shows the comparison of testing accuracies of 5 methods.

Comparing Tables 2 and Table 3, it can be seen that the detection accuracies of all steganalysis methods trained on complex texture image data sets are about 5 to 7% lower than on low texture complexity image data sets. From the first four lines and the last line, we can see that the Com_Inception method proposed for complex texture images has improved the accuracy of the other four algorithms. So the proposed steganalysis method of using 30 high-pass filtering kernels and inception-like architecture can improve the detection accuracy for complex texture images.

5.3.3 The effectiveness of the whole framework

In this section we use another data set BOSSbase v0.92 to test the whole framework's effectiveness. Two trained optimal steganalysis models are obtained from section 5.3.1 and section 5.3.2, which are noted as complex model and simple model. The other 4 comparison steganalysis algorithms are trained on the whole BOSSbase v 1.01 without splitting. For each image in test set, we first calculate its texture complexity, by using the texture complexity metrics proposed in section 2.1. If the image texture is complex, we load the complex model to detect whether the image was embedded, otherwise, the simple model are loaded. For convenience, we note the steganalysis framework for different texture complexity images as SF-DTC. Table 4 compares the effectiveness of our framework with other methods.

Comparing Tables 2, 3 and 4, it can be found that the first four steganalysis methods are trained on the entire unclassified BOSSbase v1.01 data set. The detection accuracies are lower than that on the simple texture complexity images alone, but higher than on the complex texture images alone. Because in a mixed data set, simple texture images are mixed in complex texture images, the detection accuricies are between the two separate situations. However, the detection accuracy of steganalysis framework proposed in this paper remains above 90%, showing that the steganalysis framework of dividing data set according to texture complexity and training the networks separately can specially improve the final performance by detecting suspicious images.

Table 3 The comparison of testing accuracies of 5 methods in complex texture case

Steganalysis Methods	HUGO	WOW	S-UNIWARD	MiPOD
SRM + EC	82.2%	81.3%	80.5%	79.8%
Qian's network	81.5%	80.8%	80.4%	80.1%
Xu's network	83.3%	82.6%	82.1%	81.4%
DRSN	85.6%	85.2%	84.9%	84.2%
Com_Inception(our method)	86.8%	86.7%	86.5%	85.1%

Table 4 The comparison of testing accuracies of 5 methods

Steganalysis Methods	HUGO	WOW	S-UNIWARD	MiPOD
SRM + EC	84.4%	83.5%	83.1%	82.2%
Qian's network	84.5%	83.7%	84.2%	82.1%
Xu's network	86.3%	86.2%	85.8%	84.4%
DRSN	88.2%	87.5%	87.1%	86.4%
SF-DTC	92.7%	92.1%	91.4%	90.3%

5.3.4 Ensemble method

We use the bagging ensemble method to improve the detection accuracy of the whole framework. First, according to the method in Section 4.1, the BOSSbase v1.01 data set is cropped into 40,000 images, and then divided into two sets of simple texture and complex texture, each containing 20,000 images. Then taking the simple texture image set as an example, by using the bagging method, 15,000 images are randomly selected from the 20,000 images as the training set, and the remaining 5000 images are used as the verification set. Repeat 3 times in this way to get three pairs of 15,000/5000 data sets, and then carry out steganographic operations respectively to train three steganalysis networks for simple texture images. In the test phase, the "majority voting" principle is used to obtain the final classification results. The ensemble idea is shown in Fig. 12. Table 5 shows the comparison of detection accuracies of SF-DTC before and after using bagging ensemble method.

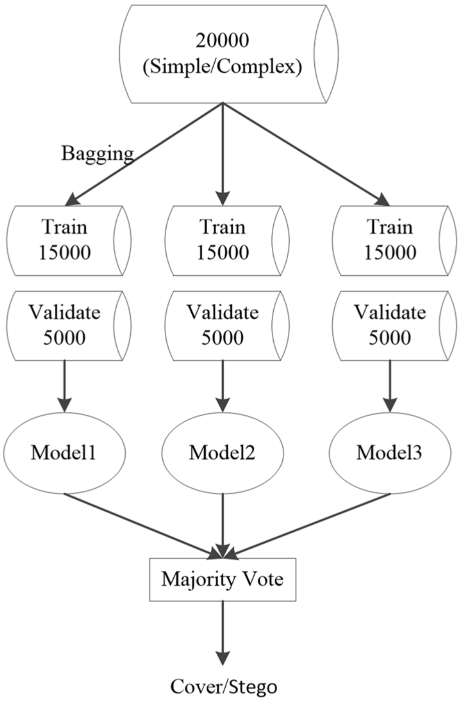


Fig. 12 The bagging ensemble method on simple/complex texture image set

Table 5 Comparison of detection accuracies of SF-DTC before and after using bagging ensemble method

Steganalysis Methods	HUGO	WOW	S-UNIWARD	MiPOD
SF-DTC	92.7%	92.1%	91.4%	90.3%
SF-DTC (bagging)	94.5%	93.4%	93.3%	92.2%

From Table 5, we can see that after using the bagging ensemble method, the detection accuracies of two steganalysis networks are improved when training models on two texture complexity image sets. So the performance of the whole SF-DTC is increased. These experimental results show the effectiveness of our proposed steganalyais framework.

6 Conclusion

Because the image texture complexity has a certain influence on steganalysis, it's not effective to mix all images with different complexities to train the steganalysis network. This paper considers the image texture complexity, and uses the contrast and homogeneity of the gray level co-occurrence matrix to determine the texture complexity of an image. Then, according to the texture complexity, the image data set is divided into two data sets: simple texture images and complex texture images. The corresponding steganalysis network is designed and trained separately for each type of images with similar texture complexities. For simple texture images, we use the MER idea to extract the most effective region of each image, and then use the most effective region for feature extraction. For complex texture images, we use 30 high-pass filters for preprocessing to enhance the steganographic signals, and adopt the "Inception" idea to extract richer features at different scales by four different sizes of convolution kernels. In the test phase, the texture complexity of each image in the test set is calculated, and a steganalysis model with the corresponding complexity is loaded and used to determine whether the test image contains secret information. Finally, when training the networks on two data sets separately, the bagging ensemble method improves the detection accuracy of the entire steganalysis framework.

However, in this paper, for each of the different adaptive steganographic algorithms, we need to use the corresponding steganographic images to train networks separately. So the network's generalization ability on other adaptive steganographic algorithms is not strong, that is, it is not the true universal steganalysis. In future, we will further study this aspect.

References

1. Amirkhani H, Rahmati M (2011) New framework for using image contents in blind steganalysis systems. J Electron Imag 20(1):013016
2. Cho S, Cha BH, Wang J et al (2013) Block-based image steganalysis: algorithm and performance evaluation. J Vis Commun Image Represent 24(7):846–856
3. Deng G, Zhao XF, Huang W, Sheng RN (2012) Image texture complexity estimation approach for steganography evaluation. Comput Eng 38(14):116–118

4. Dhanawe S, Hanchate DB (2014) Tamper detection in database using forensic analysis. Int J Adv Trends Comput Sci Eng 3(6):2319–7242
5. Dixit A, Dixit R (2017) A review on digital image watermarking techniques. Int J Image Graph Sign Process 9(4):56–66
6. Filler T, Judas J, Fridrich J (2012) Minimizing additive distortion in steganography using syndrome-trellis codes. IEEE Trans Inform Foren Sec 6(3):920–935
7. Fridrich J, Filler T (2007) Practical methods for minimizing embedding impact in steganography. Proc SPIE - Int Soc Optic Eng 6505:650502–650502-15
8. Fridrich J, Kodovsky J (2012) Rich models for steganalysis of digital images. IEEE Trans Inform Foren Sec 7(3):868–882
9. He K, Zhang X, Ren S, Sun, J (2016) Deep residual learning for image recognition. Proc IEEE Conf Comput Vision Pattern Recogn Las Vegas: 770–778
10. Holub V, Fridrich J (2013) Digital image steganography using universal distortion. ACM workshop on information hiding and multimedia security (pp.59-68). ACM
11. Holub V, Fridrich J (2012) Designing steganographic distortion using directional filters. IEEE Int Workshop Inform Foren Sec 2(4):234–239
12. Hu D, Shen Q, Zhou S, Liu X, Fan Y, Wang L (2017) Adaptive steganalysis based on selection region and combined convolutional neural networks. Sec Commun Netw 2017(4):1–9
13. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. Proc 32nd Int Conf Mach Learn Lille: 448–456
14. Juarez-Sandoval O, Cedillo-Hernandez M, Nakano-Miyatake M et al. (2016) Image-adaptive steganalysis for LSB matching steganography. Proc 39th Int Conf Telecommun Sign Process, Vienna: 478–483
15. Liu T, Wang S (2014) A steganalyis scheme based on improved local texture features. Inform Sec Commun Privacy 4:96–102
16. Liu Q, Sung AH, Chen Z et al (2008) Feature mining and pattern classification for steganalysis of LSB matching steganography in grayscale images. Pattern Recogn 41(1):56–66
17. Liu X, Deng R, Choo K K R, et al (2018) Privacy-preserving outsourced calculation toolkit in the cloud. IEEE Trans Depend Secure Comput (99):1–1
18. Luo X, Song X, Li X et al (2016) Steganalysis of HUGO steganography based on parameter recognition of syndrome-trellis-codes. Multimed Tools Appl 75(21):13557–13583
19. Ma Y, Luo X, Li X, et al (2018) Selection of rich model Steganalysis features based on decision rough set α-positive region reduction. IEEE Trans Circ Syst Video Technol (99): 1–1
20. Mao JF (2010) Shangrao: A novel general Steganalysis technique based on cover image describing. Chin J Comput 33(33):569–579
21. Pevný T, Filler T, Bas P (2010) Using high-dimensional image models to perform highly undetectable steganography. Lect Notes Comput Sci 6387:161–177
22. Pibre L, Pasquet J, Ienco D, Chaumont M (2016) Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch. Electron Imag 4(8):1–11
23. Qian Y, Dong J, Wang W, Tan T (2015) Deep learning for steganalysis via convolutional neural networks. SPIE Med Watermark Sec Foren 9409
24. Sedighi V, Cogranne R, Fridrich J (2015) Content-adaptive steganography by minimizing statistical detectability. IEEE Trans Inform Foren Sec 11(2):221–234
25. Sedighi V, Cogranne R, Fridrich J (2016) Content-adaptive steganography by minimizing statistical detectability. IEEE Trans Inf Forensic Sec 1(2):221–234
26. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S et al (2015) Going deeper with convolutions. Proceedings of the 28th IEEE conference on computer vision and pattern recognition, Boston 1–9
27. Tang W, Li H, Luo W, Huang J (2014) Adaptive steganalysis against WOW embedding algorithm. Proc 2nd ACM Workshop Inform Hiding Multimed Sec. Salzburg, 91–96
28. Wang R, Xu MK, Ping XJ, Zhang T (2014) Steganalysis of spatial images based on segmentation. Acta Automat Sin 40(12):2936–2943
29. Wang R, Xu M, Ping X et al (2015) Steganalysis of JPEG images by block texture based segmentation. Multimed Tools Appl 74(15):5725–5746
30. Wang LN, Tan XZ, Xu YB, Zhai LM (2017) An adaptive JPEG image steganography based on texture complexity. J Wuhan Univ 63(5):421–426
31. Wu S, Zhong S, Liu Y (2018) Deep residual learning for image steganalysis. Multimed Tools Appl 77(9): 10437–10453
32. Wu S, Zhong S, Liu Y (2017) Residual convolution network based steganalysis with adaptive content suppression. IEEE International Conference on Multimedia and Expo. pp 241–246
33. Xu G, Wu HZ, Shi YQ (2016) Structural design of convolutional neural networks for steganalysis. IEEE Sign Process Lett 23(5):708–712

34. Ye J, Ni J, Yi Y (2017) Deep learning hierarchical representations for image Steganalysis. IEEE Trans Inform Foren Sec 12(11):2545–2557
35. Yuan Y, Lu W, Feng B, Weng J (2017) Steganalysis with CNN using multi-channels filtered residuals. Proc 3rd Int Conf Cloud Comput Sec, Nanjing, 110–120
36. Zhang Y, Qin C, Zhang W et al (2018) On the fault-tolerant performance for a class of robust image steganography. Signal Process 146:99–111

Shangping Zhong received his Ph.D. in computer science and technology from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. Now, he is a Professor at the College of Mathematics and Computer Science, Fuzhou University in Fuzhou, China. His research interests include machine learning, intelligent image analysis and information security.

Cunmin Jia received his B.S. degree in information security from college of mathematics and computer science, Fuzhou University, Fuzhou, China, in 2016, where he is currently pursuing the M.S. degree. His research interests include image steganalysis, machine learning and deep learning.

Kaizhi Chen received his Ph.D. degree in information and communication engineering from Southeast University, Nanjing, China, in 2011. His research interests include biometric features recognition and information security.

Peng Dai received his B.S. degree in information security from School of computer engineering, Qingdao University of technology, Qingdao, China, in 2016. Now he is pursuing the M.S. degree in Fuzhou University, Fuzhou, China. His research interests include image processing, machine learning and deep learning.

Affiliations

Shangping Zhong[1,2] • **Cunmin Jia**[1,2] • **Kaizhi Chen**[1,2] • **Peng Dai**[1,2]

[1] College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China

[2] University Key Laboratory of Information Security of Network Systems (Fuzhou University), Fuzhou 350116 Fujian Province, China